

L2 INFORMATIQUE
Groupe B
Année 2015

Application en Java pour la gestion en local des connaissances liées à un sujet

LANIESSE Shyaka ANSELME Guenael ALIGNAN Alain

Tuteur de projet : Abdelhak-Djamel Seriai

Année universitaire 2014-2015

Chapitre 1

Introduction

Nous sommes un groupe de 3 étudiants, Guenael ANSELME, Alain ALIGNAN, et Shyaka LANIESSE. Nous trouvons chacun sans groupe nous avons décidé de former le notre car on s'entendait assez bien sans trop se connaître.
On peut définir notre projet comme une banque de connaissances en local. Notre application s'intéresse à l'organisation de connaissances pouvant se rapporter à un sujet précis.
Par "connaissances" nous définissons en fait tous les **documents** pouvant se rapporter au sujet désigné.
Il fallait donc pouvoir jongler entre une interface graphique, pour l'application, et une base de données contenant les fichiers liés aux sujets.
A partir de là, plusieurs tâches pouvaient être extraites mais nous en verrons les détails dans une prochaine partie de ce rapport.

Pédagogiquement, ce projet servirait à gérer nos documents de cours afin de pouvoir tout retrouver facilement sans passer par l'explorateur de fichiers.
A mon plus grand regrêt nous avons démarré le projet tardivement. En effet nous n'étions pas tous autant impliqués dans ce projet ce qui a fait que j'ai (Shyaka) dû prendre en main la commencement et l'avancement du projet.
En ce qui concerne le travail, nous avons procédé à la mise en place d'une SVN car nous préférions travailler chacun chez soi et opérer en ligne.

Le déroulement du projet se présente comme ceci :
- Réalisation des croquis de l'interface graphique, diagramme UML de l'application
- Implémentation de l'interface graphique + Réalisation de la base de données [1]
- Implémentation du lien Java - Base de données
- Finalisation de l'interface graphique

J'ai choisi d'organiser chaque chapitre en fonction de ce déroulement afin que l'on retrouve toujours le même fil conducteur.
Je vous laisse maintenant plonger dans l'histoire du développement de cette application.

Sommaire

Chapitre 2

Présentation du domaine et des aspects techniques : une banque de connaissance

Notre application concerne l'organisation des connaissances nous pouvons donc affirmer qu'elle s'insère dans le domaine de la représentation, le stockage et la gestion des connaissances et donc fait référence à des banques de connaissances.

Comme définition générale, nous pouvons dire que la représentation des connaissances correspond à modéliser un ensemble de données afin de les rendre plus facilement consultables et manipulables. Le stockage lui permet juste la mise en réserve des connaissances souhaitées dans un support (ici dans une base de données). Ajouter à ça la gestion des connaissances est l'organisation de ces dernieres dans notre application donc à la fois dans notre interface et notre base de données. L'approche object peut être utilisée pour ce domaine, c'est la cas pour ce projet ou l'on utilise un langage orienté objet, le Java.
Lorsque l'on utilise une base de données, ces connaissances sont placées à l'intérieur et pour les gérer on utilise un Système de Gestion de Base de Données (SGBD).

Enfin, il existe souvent des cas où la SGBD ne suffit pas, il faut quelque chose en plus pour l'organisation des connaissances. C'est par exemple le cas lorsque l'on veut partager ces connaissances (en ligne ou en local) ou tout simplement si l'on veut une gestion plus "intuitive" avec une interface graphique (ou autre).

Notre application s'implique bien dans cette définition ce qui nous reste à définir les caractéristiques précises du domaine dans lequel elle se situe.

2.1 La base de données

Pour faire simple, une base de données peut être définie comme une entité (un "conteneur") dans lequel on stock des informations (chiffres, mots, dates, fichiers..) de manière structurée. Ces informations sont accessibles et manipulées à l'aide d'un SGBD qui rend l'utilisation d'une base de données accessible avec un peu de conviction.
Ces informations sont donc organisées et forment une table. Une table correspond au domaine principal auquel sont rattachées les informations contenues dans la table.
Parmi les tables les informations peuvent être retraitées par des opérations de recherches, de tris, de jointures..

Parmi les SGBD disponibles nous avons choisi MySQL [2] un des plus répendu et utilisant comme vous l'avez sans doute compris le langage SQL.
C'est par l'intermédiaire de ce langage que l'on peut effectuer toutes les sortes d'opérations disponibles sur les tables.

L'utilisation d'une base de données pour notre application était indispensable. Il nous fallait un système pour déposer les documents mais aussi stocker nos sujets eux-mêmes.
Le fait de pouvoir faire des opérations de recherches/tris/jointures correspondait exactement à nos attentes afin de pouvoir associer par exemple un sujet à ses sujet connexes mais nous verrons tout ceci en détails par la suite.

2.2 La programmation Java

Nous arrivons maintenant dans le domaine de la programation à proprement parler. On nous a demandé une application en Java, je me suis donc demandé pourquoi en Java [3] alors en voici les quelques aspects intéréssant.

Java est un langage orienté objet ce qui veut dire qu'il est constitué d'un ensemble de structures de données (objets). Ce dernier est lui-même constitué de données et de méthodes qui, elles aussi, manipulent des données.
Ce qui est intéressant par rapport à notre application est que java est un programme portable. C'est-à-dire que le code du programme créé pourra être utilisé dans divers environnements, il n'est pas destiné à être exécuté directement par le système d'exploitation.

L'utilisation de Java est donc parfaite pour une application en local. On veut que chacun puisse l'utiliser facilement peu importe son système d'exploitation juste avec une simple compilation ! Qui plus est, Java nous permet de développer notre application sous forme de fenêtres ce qui est grandement utile pour faciliter l'utilisation de l'application.

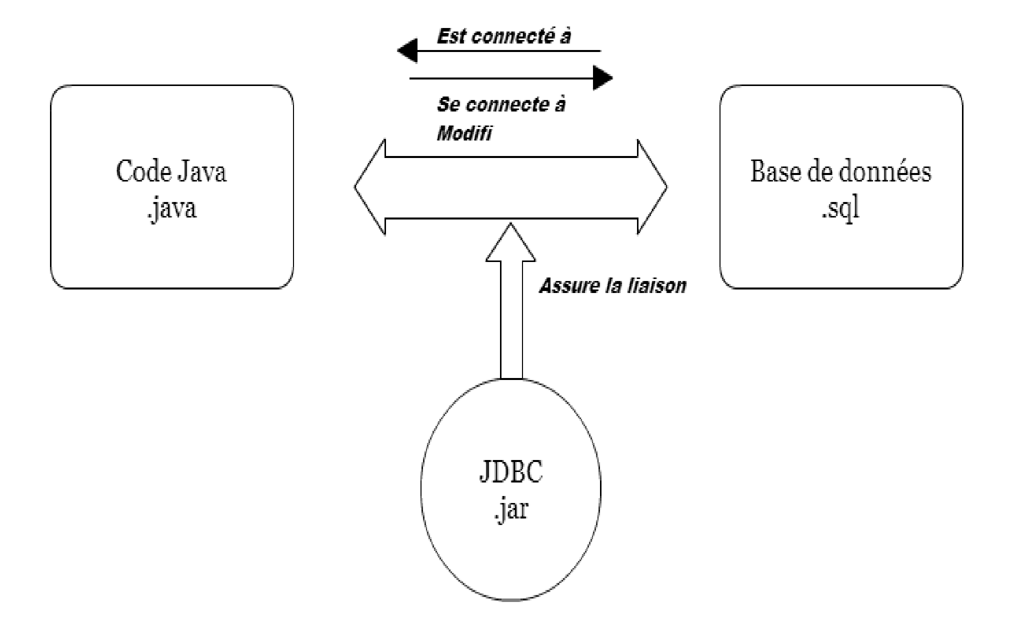
2.3 Joindre les deux bouts..

Nous avons la base de données, notre programmation Java mais comment les faire communiquer ?

En théorie c'est tout simple ; il s'agit en fait d'un JDBC (Java DataBase Connectivity). Ce dernier contient des classes Java nous permettant de nous connecter à la base de données et effectuer les opérations dessus dans notre code Java. On télécharge donc des pilotes JDBC (dans un fichier .jar), on les inclut dans notre projet Java (dans le CLASSPATH pour être tranquile) et hop, après quelques petites lignes de routine on se connecte à notre base de données et on lui envoit les requêtes que l'on veut.
On se retrouve un peu avec un langage inclus dans un autre puisque nos requêtes sont toujours écrites en SQL mais dans notre code Java.

Je vous ai placé ci-dessous un schéma simplifié d'utilisation des différents acteurs menant à bien le déroulement de l'application.

SCHEMA D'UTILISATION



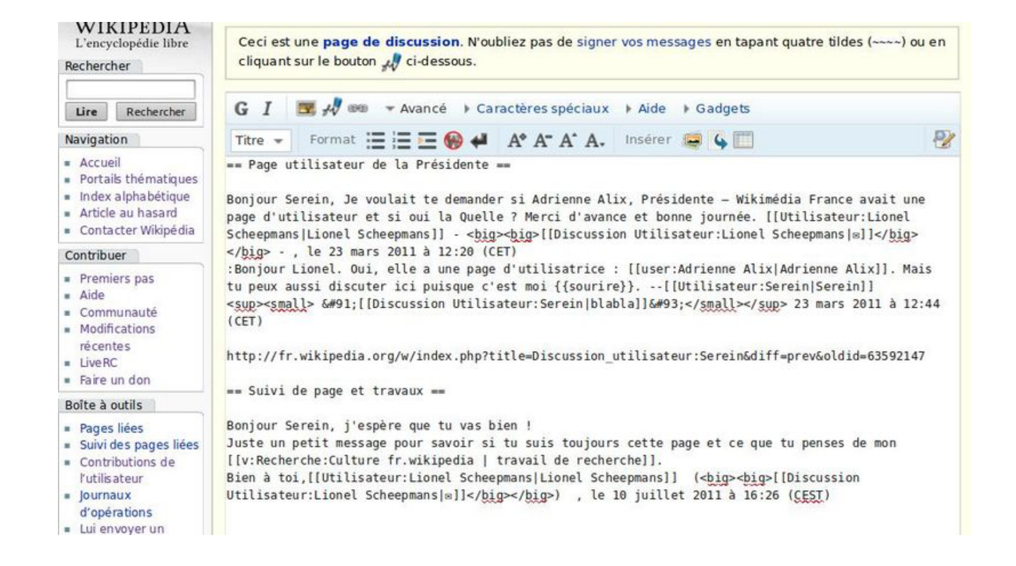
Chapitre 3

Présentation du problème

3.1 Etude de l'existant

Avant de démmarer une application, il faut faire une étude de l'existant afin de pouvoir s'orienter dans le développement de son projet. Orientés par le titre de notre projet et après avoir fait des recherches sur internet, on peut affirmer que notre projet se rapproche fortement de ce que l'on appelle des Wikis, des banque de connaissances.

Concernant l'analyse, tout d'abord un Wiki est une documentation en ligne destinée à la création et la modification de connaissances par les utilisateurs par exemple Wikipedia. Un wiki utilise le plus souvent un langage de balises assez simple. Lorsque l'on accède au Wiki en écriture, pour modifier ou ajouter une page à un Wiki, la page se présente entièrement dans le navigateur web parfois sous forme de formulaires web.



Le problème résultant de mon analyse est que le wiki est d'une part en ligne et non en local ce qui est pour nous le coeur de notre application mais aussi modifiable par tous. En effet le notre ne sera pas modifiable par tous car c'est un "Wiki" personnel mais il pourrait se partager et s'alimenter grâce aux utilisateurs. De ce fait, notre application sera similaire à un wikipedia de par son utilisation (alimentation de données, mise à jour immédiate..) mais différente à cause de l'aspect de partage d'un wiki (en ligne et modifiable par tous), seul l'utilisateur peut ajouter/modifier des documents à son wiki.

3.2 Une histoire d'interface

Je vais vous décrire le tout au présent afin que vous puissiez suivre le développement de du raisonnement au mieux.

A la lecture du sujet, la première contrainte est qu'il nous faut donc créer une application utilisable pour tous. Cela implique des personnes ne connaissant rien en programmation, on a donc besoin d'une interface graphique intuitive. Cette interface graphique doit à la fois être structurée et l'application agréable à parcourir.Il faut réfléchir à une structure de page commune aux différentes fenêtres que pourra présenter l'application. Le plus gros problème est que Java est un langage inconnu (merci [4] et [5]) et qu'il va donc falloir se plonger dedans pour développer cette interface et la structurer.

Ceci étant fait, le problème est de pouvoir naviguer à travers les différentes fenêtres. C'est là qu'interviennent les objets tels que les boutons. Pour pallier à ce problème, on choisit de faire un menu composé de boutons, ces derniers redirigeraient vers les fenêtres souhaitées.

Pour finir cette interface, il faut aussi ajouter quelques zones de textes, faire des boutons sur le panneau principal de la page (voir section 2.2). Un autre problème se présente, celui de savoir comment matérialiser la fenêtre d'ajout/-modification d'un sujet. En effet, cela pourrait être une nouvelle fenêtre, un onglet, une boîte de dialogue..

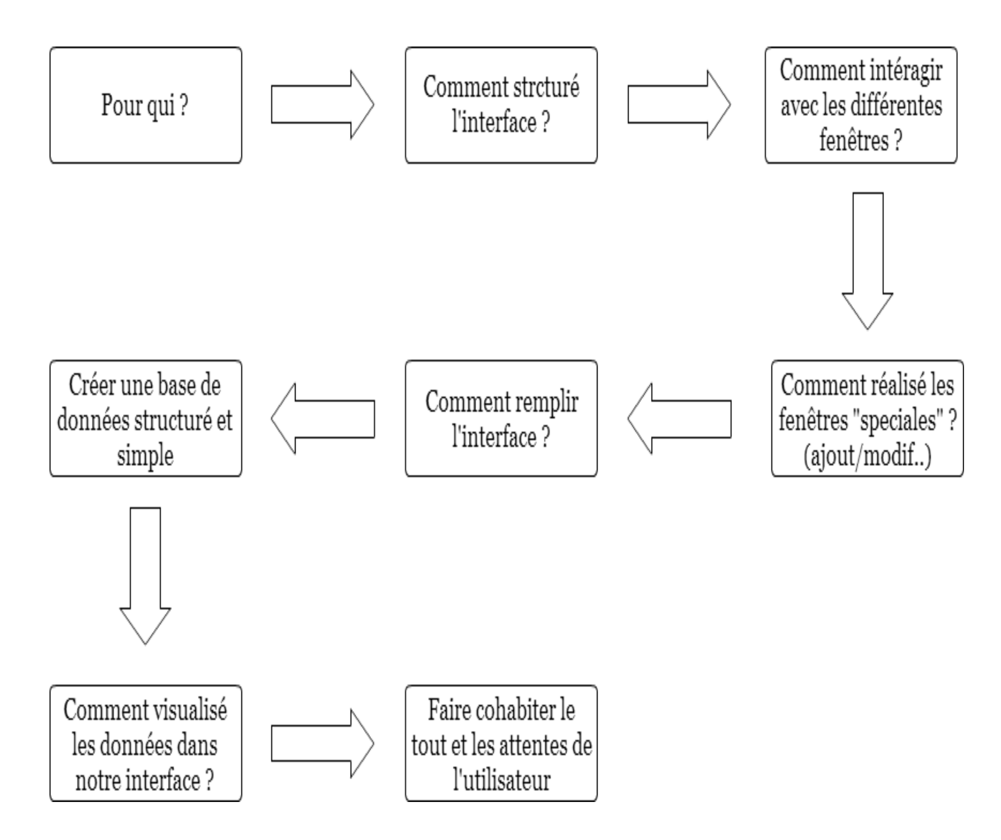
3.3 La gestion et visualisation des données

Une fois l'interface créée, elle nous parrait un peu vide.. Pour la remplir il faut s'attaquer en premier lieu à la création de la base de données du projet. La base de données doit respecter les règles suivantes :

1. La base de données doit pouvoir héberger les documents avec une taille limite.
2. Elle doit être assez simpliste pour que les opérations de modification soit optimisées
3. Les tables contiendront le moins de champs possibles pour la clarté de la base de données
4. On doit joindre les tables entres elles pour éviter la surcharge

La base de données réalisée, il faut maintenant pouvoir visualiser les données des tables que l'on veut au sein de l'interface graphique.
Il faut aussi pouvoir gérer la base de données à la création/modification d'un sujet via l'interface graphique et donc via des petits objets comme les boutons.

L'implémentation du tout nous enmène vers de nouveaux problèmes liés à l'interface graphique et pour les résoudre, il nous faut modifier notre interface graphique initiale afin qu'elle puisse répondre aux attentes de l'utilisateur mais aussi aux demandes internes de gestion.
Comme un schéma est toujours plus agréable en voici un représentant les problèmes précis ammenés par la réalisation de cette application.



Chapitre 4

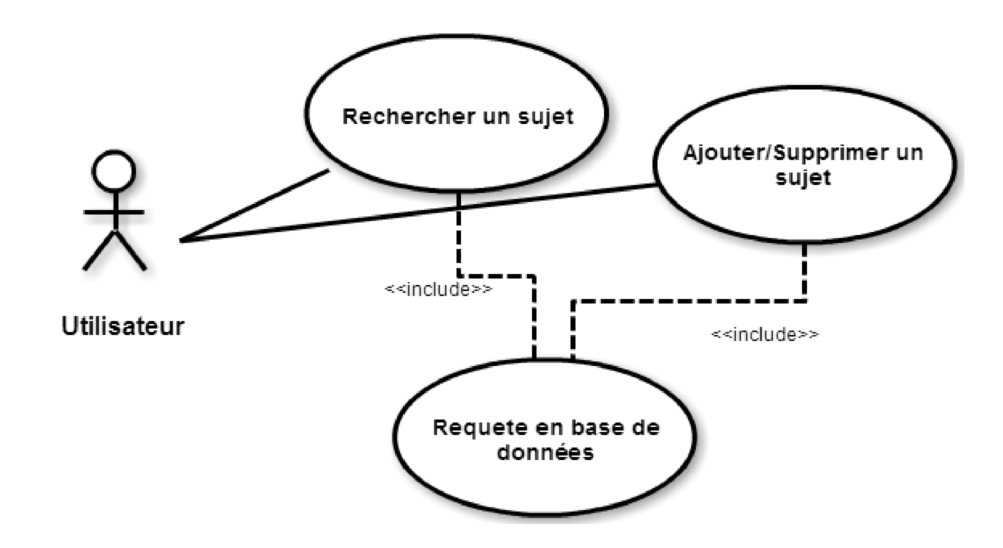
Description du travail réalisé

4.1 Analyse et conception

Après avoir cerné le problème, il a fallu le modéliser. Pour ce faire j'ai choisi de faire un diagramme de cas d'utilisation d'abord afin de définir ce que l'utilisateur serait capable de faire avec l'application.
Dans un premier temps l'utilisateur doit être capable d'ajouter un sujet, le modifier et le supprimer. En effet ceci constitue la partie principale du stockage de connaissances.

D'un autre côté, il doit aussi pouvoir trouver un sujet ou une catégorie à l'aide d'une simple recherche.
Ces deux utilisations de l'application exécuteront des requêtes sql modifiant notre base de données.

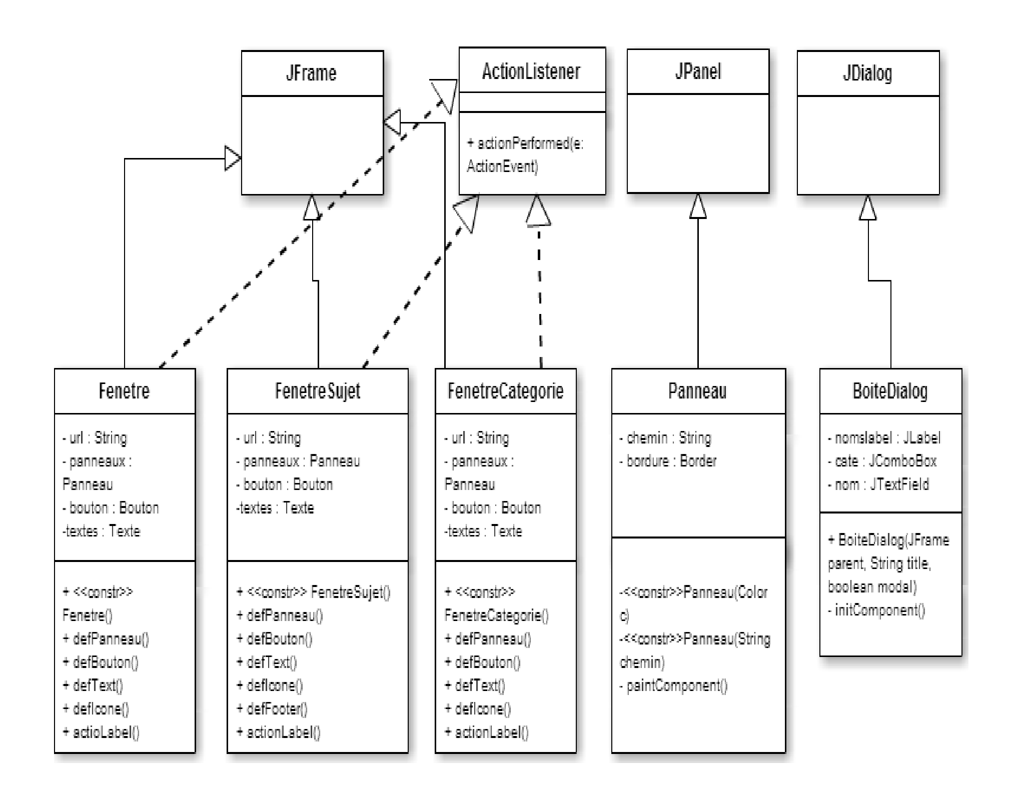
Ci dessous le Diagramme.



Après le diagramme d'utilisation j'ai décidé de définir des croquis pour l'interface graphique. Ces croquis ne seraient pas définitifs puisque durant le développement je pourrais avoir envie de changer un aspect quelconque de l'interface. Pour l'interface j'ai choisi un design simple et classique pour débuter afin que l'application soit intuitive pour l'utilisateur (menu, pied de page, bouton en haut de page..) mais bien évidemment étant débutant je savais que cela allait me demander beaucoup de travail. Vous pouvez voir mes croquis de départ en annexe.

En m'intéressant de plus près au Java j'ai vu que mon code allait être plus long que ce que je pensais. J'ai donc décidé de faire un petit diagramme de classes afin de ne pas me perdre pendant l'implémentation. Sur le diagramme ne sont pas représentés :

1. La classe FenetreContact qui a la même forme que la classe Fenêtre mis à part le panneau central.
2. La classe Recherche ouvrant juste une boite de Dialogue demandant d'entrer un mot et affichant les résultats équivalents au mot.
3. Les classes Texte, Panneau, Bouton respectivement *"extend"* JLabel, JPanel, JButton qui ne font que définir ces objets.



Au début je n'avais pas les classes JFrame, JPanel, JDialog et ActionListener dans mon diagramme de Classes car je ne les connaissais pas, tout simplement c'est en commençant à coder que je les ai découvertes et donc rajoutées dans mon diagramme en faisant hériter mes classes de celles prédéfinies.

Les méthodes *defPanneau(), defTexte(), defBouton(), defIcone()* définissent pour chaque classe le Panel, les zones de textes, les boutons, et les images du pied de page mais nous verrons cela plus en détails dans la partie suivante cependant c'est grâce à ces méthodes que tous nos composants seront bien placés et efficaces. La méthode *actioLabel()* indique comment nous allons accéder au lien lors du clic sur les JLabels qui sont ici nos icônes du pied de page(facebook, googlePlus..). J'ai trouvé que c'était plus intéressant de faire comme cela que de faire des méthodes pour chaque partie de l'interface (centre, menu, bas).

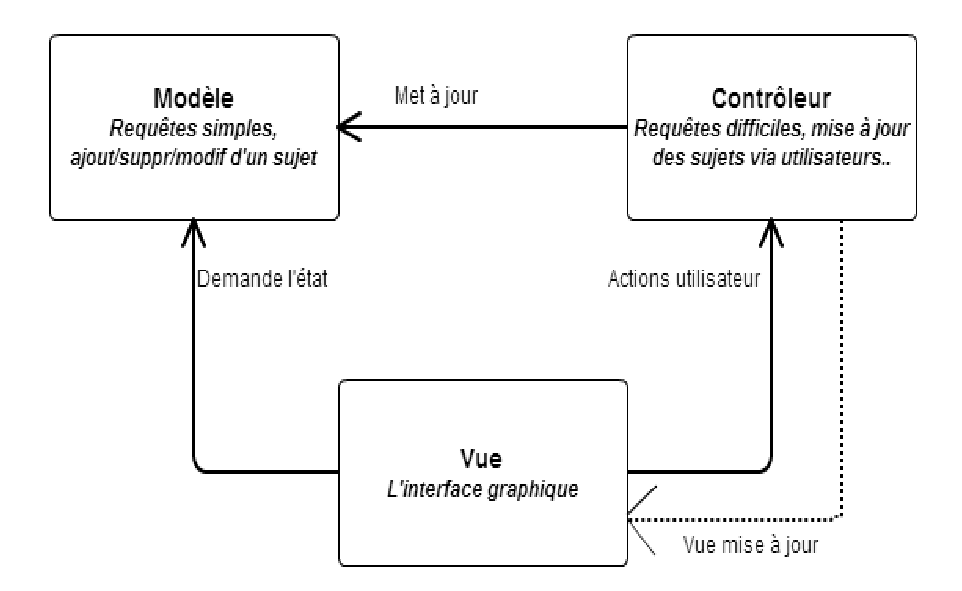
Enfin, en réalité les attributs des classes FenetreX sont multiples, il existe donc plusieurs url pour les liens, plusieurs boutons pour les ajouts, modifications.. plusieurs textes pour les différentes zones où insérer du texte..

Maintenant que tout est prêt, on peut se plonger dans le développement, l'implémentation de notre application.

4.2 Un petit mot sur le MVC

Avant de démarrer la partie sur la réalisation du projet je me dois de vous parler du MVC (Modele View Controler).
Le MVC a trois composantes :

1. Le modèle : il correspond au code de notre application, c'est là où l'intéraction avec la base de données a lieu. Il nous présente des méthodes à utiliser pour gérer la base de données avec des requêtes de suppression/ajout/modification.
2. La Vue : C'est là que les données des requêtes sont affichées mais aussi que les demandes graphiques de l'utilisateur sont gérées (clic de souris, entrer au clavier ..).
3. Le Contrôleur : C'est lui qui met à jour la vue ou le modèle. Toute action nécessitant une modification des données est gérée par le contrôleur. Les demandes précises de l'utilisateur sont donc inscrites ici.



En effet ce pattern est modélisé dans mon projet comme ceci : Le modèle est mon code Java, il contient des méthodes exécutant quelques simples requêtes pour mettre à jour ou non ma base de données comme par exemple l'insertion d'un nouveau sujet ou la suppression d'un sujet.

Le view correspond à mon interface graphique, le répère de l'utilisateur. Il sera capable d'alimenter la base de données en connaissances mais aussi de la gérer via des intéractions avec la souris et le clavier.

Enfin le contrôleur exécute les requêtes les plus difficiles c'est-à-dire la demande à l'utilisateur de rentrer des informations qui seront à leur tour entrées en base de données ou alors effectuer un tri dans notre base de données puis en afficher les résultats..

4.3 Réalisation

Avant de commencer l'implémentation, comme je l'ai mentionnée avant, j'ai consulté quelques tutorials de Java en ligne afin de savoir un minimum comment cela allait se passer. J'ai donc décidé de commencer par l'interface graphique. Pour le fond, j'ai décidé de divisier la fenêtre en trois parties indépendantes avec trois panneaux différents placés à l'aide d'un *BorderLayout()* très adapté à ma demande ; un à gauche pour le menu, en bas pour le bas de page et le panneau central. J'ai donc appris à manipuler la classe JPanel et JFrame. Mais tout d'abord voici ma définition de ma classe panneau sans l'insertion de l'image :

```
public class Fenetre extends JFrame implements ActionListener {
        private Panneau centre = new Panneau(new Color(255,255,255,0));
        .. //pareil pour menu et piedpage
        private Panneau fond = new Panneau("images.jpg");
        ..
 public void defPanneau() {
        //taille des differents panneaux
                menu.setPreferredSize
                        (new Dimension(150, centre.getHeight()));
                piedpage.
                        (new Dimension(20, piedpage.getHeight() + 120));

                centre.setLayout(null);
                menu.setLayout(null);
                piedpage.setLayout(null);

                this.getContentPane().
                        setLayout(new BorderLayout());
                this.getContentPane().
                        add(centre, BorderLayout.CENTER);
                ..// pareil avec WEST pour menu et SOUTH pour piedpage
}
```

Après cela, j'ai défini mes classes et Bouton, Texte afin de ne pas avoir à chaque fois à définir un JLabel ou un JButton mais aussi à définir un style commun pour tous les boutons et tous les textes que je pourrais changer pour des exceptions.
Je me suis donc intéressé aux classes JLabel et Jbutton, j'ai regardé les faq et appris à les utiliser afin de créer mes deux petites classes mais aussi pouvoir les manipuler à l'interieur de mes classes Fenetre.

Maintenant que les panneaux sont définis je me suis lancé dans l'insertion du texte et des boutons. Les textes et les boutons sont relativement simples ("menu", "sujet connexes", rechercher..). Il fallait juste les ajouter au panel puis les placer j'ai donc utilisé la méthode setBounds() afin de pouvoir les placer où je voulais et de modifier en cas de besoin.

```
public void defText(){

        centre.add(textAccueil);
..
        textAccueil.setBounds(40,50, 200, 15);
..
            }

public void defBouton(){
```

```
acc.addActionListener(this);
..
textMenu.setBorder(paddingBorder);
menu.setLayout
        (new BoxLayout(menu, BoxLayout.PAGE_AXIS));
menu.add(acc);
..
centre.add(rech);
rech.setBounds
(decalage1+360, centre.getHeight()+decalageBordure, 110, 25);
}
```

Comme vous pouvez le voir j'ai utilisé *addActionListener()* afin de pouvoir activer les boutons et leur action sera définie plus loin dans le `ActionPerformed()` Ici j'ai utilisé un BoxLayout() afin de pouvoir aligner les boutons du menu en colonne. Après avoir découvert que je pouvais rendre un JLabel cliquable plutôt facilement, j'ai longuement hésité entre mettre des boutons ou des textes pour le menu mais pour l'instant j'ai laissé mon premier choix qu'était les boutons cependant je me laisse le choix de changer avant la mise en pratique de l'application.

Pour le pied de page j'ai choisi de placer des images qui, lors du clique, redirige vers la page voulue. Pour ce faire j'ai défini une méthode *actioLabel()* prenant une image (JLbael) en paramètre et un chemin du site (String).
J'ajoute ensuite un *mouseListener()* et je définis ma méthode *mouseClicked()* pour la redirection.

```
public void actioLabel(JLabel icon, String site){
 icon.addMouseListener(new MouseListener() {

 public void mouseClicked(MouseEvent e) {

 String url = site;

 if(Desktop.isDesktopSupported()){
        Desktop desktop = Desktop.getDesktop();
   try {
                desktop.browse(new URI(url));
                        }
                catch .. // erreurs gerer ensuite
```

Voilà pour le plus important de ma classe FenetreAcceuil, concernant les autres elle diffèrent seulement de celle-ci par leur panneau central. Par exemple pour la classe FenetreCategorie, elle affichera la liste des catégories présentes dans la base de données mais ceci viendra après dans le développement de l'application.

Après coup, j'ai décidé d'implémenter ma boite de dialogue pour l'ajout d'un sujet. Cette boite nous demande le nom, les sujets connexes et la catégorie d'un sujet avant de l'ajouter à la base de données via le bouton "OK". Je mettrais qu'une petite partie des codes (ceux qui me semblent les plus importants, le reste se trouvant en annexe).
Mais aussi le panneau du nom et celui des sujets connexes et la description étant semblables je ne les mettrais pas ci-dessous. Avant tout il a fallu ajouter l'action au bouton "Ajout d'un sujet" dans la classe FenetreX on va ensuite s'intéresser à la méthode *initComponent()* :

```
public void actionPerformed(ActionEvent arg0) {
        if(arg0.getSource() == add){
                BoiteDialog bd =
                new BoiteDialog(null, "Ajout d'un sujet", true);
        }
}

public class BoiteDialog extends Jdialog{
.. // atributs
public BoiteDialog(
        JFrame parent, String title, boolean modal){
        ..
  this.initComponent();
}
privade void initComponent(){

 JPanel panNom = new JPanel(); // pour tous les sous panneaux
 ..
 nom = new JTextField();
 panNom.setBorder
        (BorderFactory.createTitledBorder("Nom du sujet"));
 ..
 cate = new JComboBox();
 cate.addItem("Programmation");
 ..
 content.add(panNom); // pareil pour autre panneau
 .. // creation bouton ok et annulez
        }
}
```

Passons maintenant à la base de données, j'ai choisi d'utiliser mysql. La table de données étant assez simple et presque vide au départ il suffit juste pour nous de créer les tables nécessaires. J'ai choisi une table sujet et une sujetco. En effet dans la table sujet nous mettrons les fichiers, la date et l'id du sujet. Cet id sera le même que l'id(clef primaire) de la table sujetco qui, elle, contiendra la catégorie, le nom du sujet et les sujets connexes. Ci-dessous la création de la

table Sujet :

```
CREATE TABLE sujet (
   id INT UNSIGNED NOT NULL AUTO_INCREMENT,
   id_sujet INT NOT NULL,
   url MEDIUMBLOB DEFAULT NULL,
   date DATE DEFAULT NOT NULL,
  PRIMARY KEY (id)
)
```

La dernière partie de l'implémentation consistait à faire le lien entre la base de données et le code Java. Ici encore j'ai dû prendre du temps pour apprendre. J'ai donc téléchargé le pilote JDBC + mysql facilement trouvable sur le net et l'ai introduit dans mon projet afin de pouvoir l'utiliser. J'ai commencé par le plus simple c'est-à-dire, afficher les catégories existantes dans la fenêtre catégories.

```
ResultSet result =
 state.executeQuery
  ("SELECT DISTINCT categorie FROM sujetco");
```

J'ai ensuite choisi de faire une boucle et d'ajouter chaque résultat de la requête dans un JLabel pour le placer dans le panneau central :

```
while(result.next()){
 String nomC = result.getString(1);
 Texte nameC = new Texte(nomC);
 centre.add(nameC);
 nameC.setBounds(75,100+i , 105, 15);
 i = i+30;
         }
```

Le même genre de procédé est utilisé pour l'affichage des sujets connexes ou le résultat d'une recherche. Une des fois où cela diffère est pour l'ajout d'un sujet où il a fallu créer plusieurs variables récupérant les informations rentrées par l'utilisateur avant de les rentrer en base.

```
String nomSujet = nom.getText(); // Pareil pour sujetCo..
 String nomCate = cate.getSelectedItem().toString();
         ..
state.executeUpdate
 ("INSERT INTO sujetco (sujet, categorie, sujet1, sujet2, sujet3) "
  + "VALUES
    ( '"+ nomSujet +"','" + nomCate+ "','"+ nomSujet1 +"'
            ,'"+ nomSujet2 +"','"+ nomSujet3 +"' )");
```

S'achève ici la partie de réalisation de mon travail ! J'espère avoir été au plus clair en présentant ma manière de travailler, mes principales classes/méthodes..

J'aimerai finir sur le fait que j'ai acquis énormement de connaissances concernant le langage Java et les base de données surtout car j'ai démarré de rien et j'ai le sentiment d'avoir accompli quelque chose par moi même en m'étant bien creusé la tête. En effet j'ai naviguer sur de nombreux forum (Stack OverFlow, openclassroom) pour trouver les réponses à mes problèmes ou du moins des solutions qui s'en approchaient le plus mais ça n'etait pas toujours assez.

Chapitre 5

Pour conclure

5.1 Les apports et l'avenir de l'application

La réalisation de cette application a révélé une très bonne appréciation pour le langage Java, surtout la réalisation de l'interface graphique. En effet j'ai plusieurs fois mis de côté le projet en pleine programmation et réalisé d'autres programmes en parallèle !
En m'y intéressant un peu plus ceci m'a permis de consolider mon projet professionnel et personnel qu'est le développement d'applications. Je trouve que permettre à des gens de simplifier leur vie à l'aide d'une simple application est un concept plaisant !

Concernant l'avenir de l'application, je compte la diffuser à plusieurs de mes camarades pour leur gestion de documents, plus particulièrement à mes collègues de fac de droit. Évidemment je m'en servirai durant tout le restant de ma scolarité.
Par la suite j'aimerais étendre l'application de local à web (cela serait semblable à un drive) puis pourquoi pas pour mobile (Android mais surtout pas Apple !) afin qu'elle puisse être utilisable n'importe où ! De ce fait, une amélioration sera possible afin de joindre l'utile à l'agréable (animation web..).

Il pourrait s'en suivre une plateforme (en ligne) permettant à chacun de partager ses ressources mais aussi un forum afin que les gens puissent discuter dès leur gestionnaire de connaissances et partager les améliorations qu'ils voudraient voir sur l'application. La plateforme serait le plus grand apport de l'application, en effet on pourrait y trouver des bases de données toutes faites hébergées par les utilisateurs eux-mêmes facilitant l'utilisation de l'application et faisant évoluer le statut de simple utilisateur ce qui plait forcément au peuple !

Vous êtes arrivé à la fin de ce rapport, j'espère que tout a été compris, sinon n'hésitez pas à me contacter [6] et je me ferai un plaisir de vous éclairer !

5.2 Remerciement

J'aimerais remercier M. Seriai de m'avoir orienté malgré le début difficile, mais aussi de m'avoir poussé à finir ce projet au plus vite afin de pouvoir le finaliser comme il se doit avant la date limite (première ébauche du rapport à rendre 15 jours avant par exemple).

Merci également à Léah Mansouri sans qui vous vous seriez sans doute arraché les yeux à la vue du nombre incalculable de fautes d'orthographe.. Et oui je n'ai jamais été très bon en orthographe et garde un très mauvais souvenir des dictées.

Bibliographie

[1] Christian Soutou Frederic Brouard, Rudi Bruchez. *SQL (3e edition)*. Pearson Education, 2010.

[2] Chantal Gribaumont. *Administrez vos bases de donnees avec MySQL*. OpenClassroom, 2014.

[3] David Reilly. The java programming language. *Java Coffee break*, nov 1999.

[4] Cyrille Herby. *Apprenez a programmer en Java*. OpenClassroom, 2012.

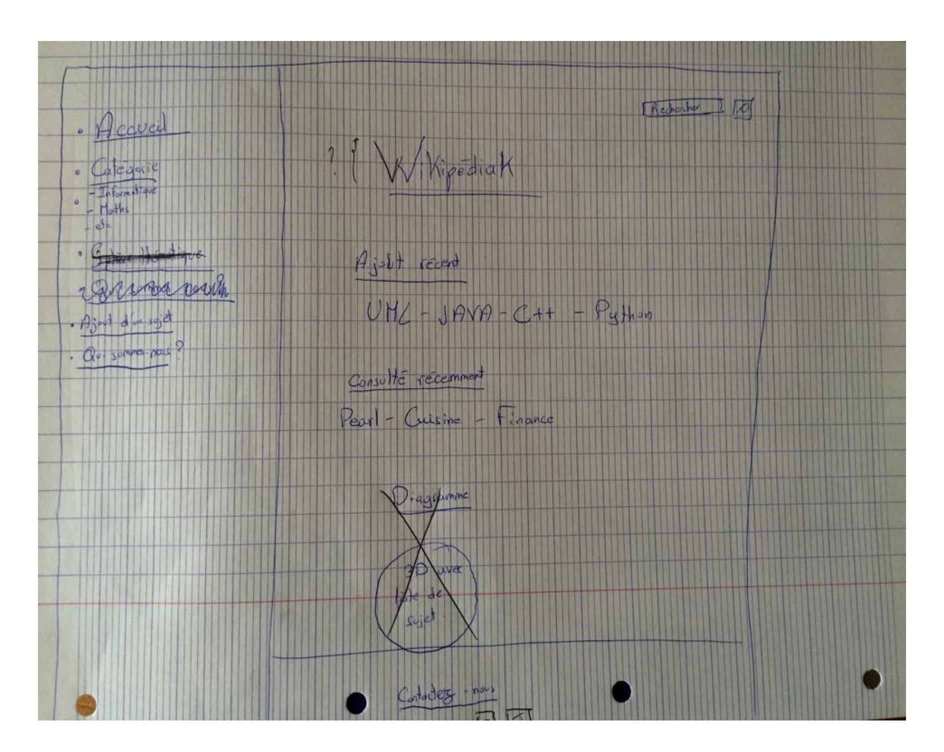
[5] Jean Francois Pillou. Comment ca marche, mar 2015. http ://www.commentcamarche.net/contents/557-java.

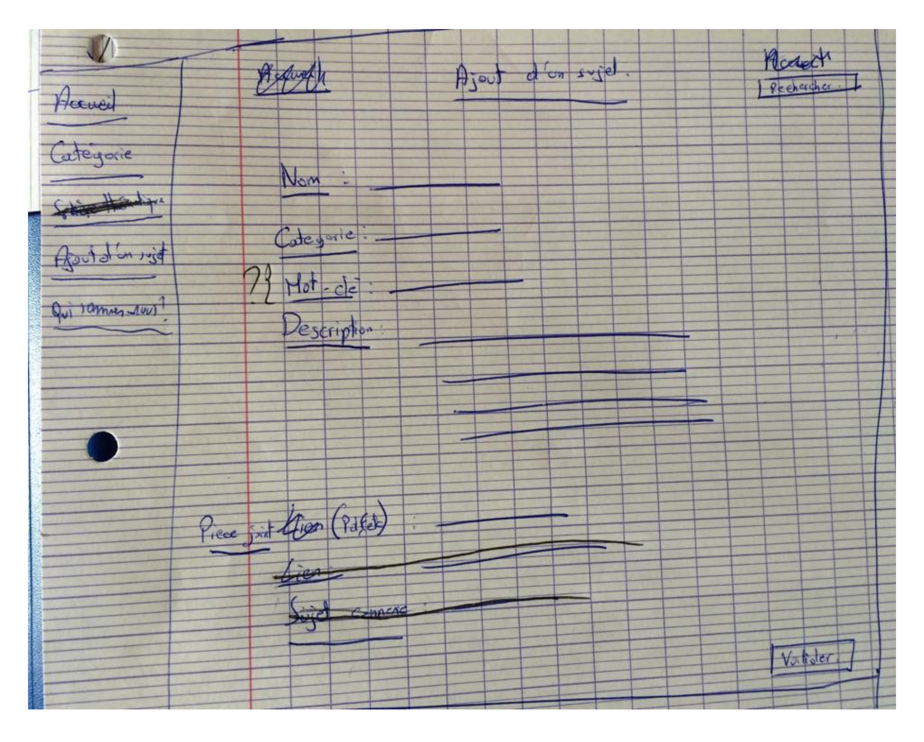
[6] Laniesse Shyaka. Rapport de projet, apr 2015. laniesse.shyaka at hotmail.com.

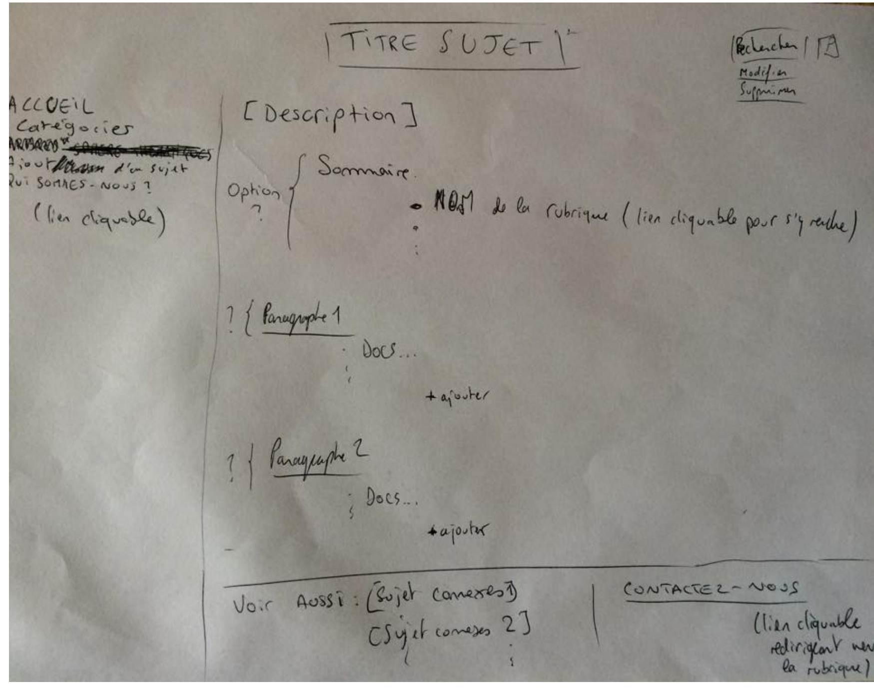
Chapitre 6

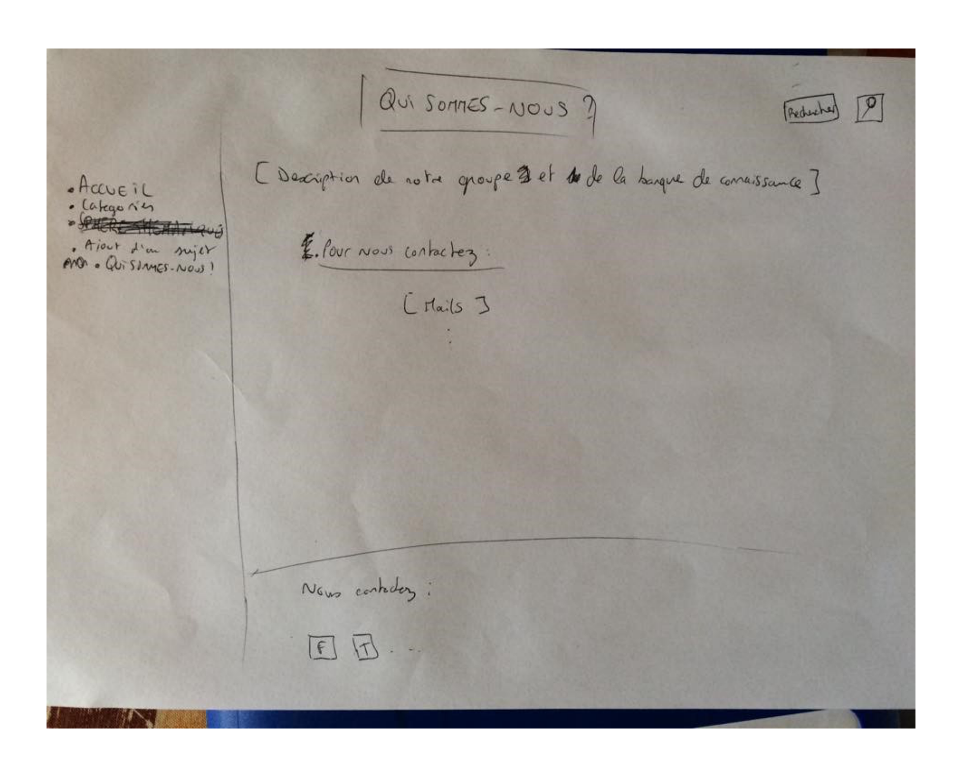
Annexe

6.1 Croquis de l'interface graphique




Ajout d'un sujet
Rechercher
Accueil
Catégorie
Ajout d'un sujet
Qui sommes-nous ?
Nom :
Catégorie :
Mot-clé :
Description :
Pièce jointe
Valider


TITRE SUJET
Rechercher
Modifier
Supprimer
ACCUEIL
Catégories
Ajout d'un sujet
QUI SOMMES-NOUS ?
(lien cliquable)
[Description]
Option ?
Sommaire
NOM de la rubrique (lien cliquable pour s'y rendre)
Paragraphe 1
Docs...
+ajouter
Paragraphe 2
Docs...
+ajouter
Voir AUSSI : [Sujet connexes 1]
[Sujet connexes 2]
CONTACTEZ-NOUS
(lien cliquable redirigeant vers la rubrique)



6.2 Quelques bout de code

Class Bouton :

```java
package projet;


import javax.swing.ImageIcon;

public class Bouton extends JButton {
  private String name;
  private Image img;

  public Bouton(String nom){
        super(nom);
        this.name = nom;
  }


  public void paintComponent(Graphics g){
        Graphics2D g2d = (Graphics2D)g;
        GradientPaint gp = new GradientPaint(0, 0, Color.white, 0, 20, Color.black, true);
        g2d.setPaint(gp);
        g2d.fillRect(0, 0, this.getWidth(), this.getHeight());
        g2d.setColor(Color.LIGHT_GRAY);
        g2d.drawString(this.name, 10, (this.getHeight() / 2) + 5);
     }
   }
```

Class Texte :

```java
package projet;

import javax.swing.JLabel;

public class Texte extends JLabel {
        private String text;

        public Texte(String texte){
            super(texte);
            this.text=texte;
            this.setFont(new Font("TimesCourier", Font.BOLD, 12));
}
        public Texte(){
            this.text="";
            this.setFont(new Font("TimesCourier", Font.BOLD, 12));
        }
}
```

Class BoiteDialog :

```java
public BoiteDialog(JFrame parent, String title, boolean modal){
  super(parent, title, modal);
  this.setSize(550, 550);
  this.setLocationRelativeTo(null);
  this.setResizable(false);
  this.setDefaultCloseOperation(JDialog.DO_NOTHING_ON_CLOSE);
  this.initComponent();
}

private void initComponent(){


  //Le nom
  JPanel panNom = new JPanel();
  panNom.setBackground(Color.white);
  panNom.setPreferredSize(new Dimension(220, 60));
  nom = new JTextField();
  nom.setPreferredSize(new Dimension(100, 25));
  panNom.setBorder(BorderFactory.createTitledBorder("Nom du sujet"));
  nomLabel = new JLabel("Sujet :");
  panNom.add(nomLabel);
  panNom.add(nom);

  //La catégorie
  JPanel panCate = new JPanel();
  panCate.setBackground(Color.white);
  panCate.setPreferredSize(new Dimension(220, 60));
  panCate.setBorder(BorderFactory.createTitledBorder("Catégorie du Sujet"));
  cate = new JComboBox();
  cate.addItem("Programmation");
  cate.addItem("Cours");
  cate.addItem("Culture G");
  cate.addItem("Autres");
  cateLabel = new JLabel("Categorie : ");
  panCate.add(cateLabel);
  panCate.add(cate);
```

```java
okBouton.addActionListener(new ActionListener(){
  public void actionPerformed(ActionEvent arg0) {
      FenetreSujet alt = new FenetreSujet(nom.getText());

          String url = "jdbc:mysql://localhost:3306/projet";
          String user = "root";
          String mdp = "";
          String nomSujet = nom.getText();
          String nomCate = cate.getSelectedItem().toString();
          String nomSujet1 = sujetCo0.getText();
          String nomSujet2 = sujetCo1.getText();
          String nomSujet3 = sujetCo2.getText();

        Connection conn=NULL;

        try {

              Class.forName("com.mysql.jdbc.Driver").newInstance();

              conn = DriverManager.getConnection(url, user, mdp);

              Statement state = conn.createStatement();

              state.executeUpdate("INSERT INTO sujetco (sujet, categorie, sujet1, sujet2, sujet3) "
                    + "VALUES ( '"+ nomSujet +"','" + nomCate+ "','"+ nomSujet1 +"','"+ nomSujet2 +"','"+ nomSujet3 +"' )");


              conn.close();
              state.close();

            } catch (Exception e) {
              e.printStackTrace();
            }


      alt.setVisible(true);
      setVisible(false);
    }
});
```

Class FenetreSujet :

```java
public class FenetreSujet extends JFrame implements ActionListener {

    //url pour pied de page
    private String toitS = "https://le9emetoit.wordpress.com/";
    private String sanjaS = "http://www.besanja.com/";
    private String hurriS = "http://leshurricanes.free.fr/";
    private String fbS = "https://www.facebook.com/Jakick.QB";
    private String gplusS = "https://plus.google.com/+ShyakaLaniesse";

    private static final Connection NULL = null;
    private Panneau centre = new Panneau(new Color(255,255,255,0));
    private Panneau menu = new Panneau(new Color(255,255,255,150));
    private Panneau piedpage = new Panneau(new Color(255,255,255,0));
    private Panneau fond = new Panneau("images.jpg");

    private Bouton suppr = new Bouton("Supprimer");
    private Bouton modif = new Bouton("Modifier");
    private Bouton rech = new Bouton("Rechercher");
    private Bouton acc = new Bouton("Accueil");
    private Bouton cate = new Bouton("Catégories");
    private Bouton sphere = new Bouton("Sphère");
    private Bouton add = new Bouton("Ajout d'un sujet");
    private Bouton contact = new Bouton("Contactez-nous");

    private Texte textFooter = new Texte("Sujet Connexe :");
    private Texte textMenu = new Texte("Menu :");
    private Texte textAccueil = new Texte("");
    private Texte textFooter1 = new Texte("Mes amis :");
    private Texte textFooter2 = new Texte("Mes réseaux :");
    private JLabel fb, toit, sanja, hurri, gplus;

    private final int decalageBordure = 2;
    private final int decalage1 = centre.getWidth()+decalageBordure;
    private Border paddingBorder = BorderFactory.createEmptyBorder(10,10,10,10);
     //
```

```java
public class FenetreSujet extends JFrame implements ActionListener {

    //url pour pied de page
    private String toitS = "https://le9emetoit.wordpress.com/";
    private String sanjaS = "http://www.besanja.com/";
    private String hurriS = "http://leshurricanes.free.fr/";
    private String fbS = "https://www.facebook.com/Jakick.QB";
    private String gplusS = "https://plus.google.com/+ShyakaLaniesse";

    private static final Connection NULL = null;
    private Panneau centre = new Panneau(new Color(255,255,255,0));
    private Panneau menu = new Panneau(new Color(255,255,255,150));
    private Panneau piedpage = new Panneau(new Color(255,255,255,0));
    private Panneau fond = new Panneau("images.jpg");

    private Bouton suppr = new Bouton("Supprimer");
    private Bouton modif = new Bouton("Modifier");
    private Bouton rech = new Bouton("Rechercher");
    private Bouton acc = new Bouton("Accueil");
    private Bouton cate = new Bouton("Catégories");
    private Bouton sphere = new Bouton("Sphère");
    private Bouton add = new Bouton("Ajout d'un sujet");
    private Bouton contact = new Bouton("Contactez-nous");

    private Texte textFooter = new Texte("Sujet Connexe :");
    private Texte textMenu = new Texte("Menu :");
    private Texte textAccueil = new Texte("");
    private Texte textFooter1 = new Texte("Mes amis :");
    private Texte textFooter2 = new Texte("Mes réseaux :");
    private JLabel fb, toit, sanja, hurri, gplus;

    private final int decalageBordure = 2;
    private final int decalage1 = centre.getWidth()+decalageBordure;
    private Border paddingBorder = BorderFactory.createEmptyBorder(10,10,10,10);
     //
```

Manuel d'utilisation :

Je vais vous présenter les principales fonctions de l'application :

Ajout d'un sujet :



Pour ajouter un sujet, cliquez sur la bouton Ajout d'un sujet présent dans la menu (voir capture d'ecran ci dessus).
Rentrez les informations demandées afin que l'application les enregistre dans la Base de données.

Suppression d'un sujet :



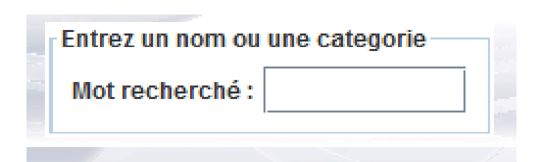
Le bouton pour supprimer un sujet se trouve en haut du panneau central de la fenêtre du sujet.
Cliquez dessus et une fenêtre de confirmation s'affichera, confirmez pour la suppression du sujet. ATTENTION vous ne pouvez pas revenir en arrière il sera définitivement supprimé avec les fichiers qu'il contient !

Modification d'un sujet :



La modification d'un sujet est similaire à l'ajout d'un sujet. Le bouton se trouve à coté du bouton "supprimer", cliquez. Une boite de dialogue s'ouvrira, la même que pour l'ajout d'un sujet, entrez les modifications que vous voulez apporter. Si vous laissez en blanc les données existantes resteront intactes.

Recherche d'un sujet :



La recherche d'un sujet se fait via le bouton "Rechercher" à côté du bouton "Modifier" en haut du panneau central. Cliquez, une boite de dialogue apparait alors avec une zone de texte. Entrez comme indiqué un mot, une catégorie ou un mot clef et il vous affichera les sujet trouvé contenant ce mot là dans ses paramètres (sujet/catégorie/sujet connexes/mot clef).